Halftoning method for the generation of motion stimuli

Jeffrey B. Mulligan and Leland S. Stone

NASA Ames Research Center, Mail Stop 239-3, Moffett Field, California 94035

Received October 24, 1988; accepted March 22, 1989

We describe a novel technique in which color table animation in conjunction with a single base image can be used to generate a broad class of motion stimuli. We have applied this technique to the generation of drifting sine-wave gratings (and by extension, sine-wave plaids). For each drifting grating, sine and cosine spatial phase components are first reduced to 1 bit/pixel by using a digital halftoning technique. The resulting pairs of 1-bit images are then loaded into pairs of bit planes of the display memory. To animate the patterns, the display hardware's color lookup table is modified on a frame-by-frame basis. Because the contrasts and temporal frequencies of the various components are mutually independent, a large number of two-dimensional stimulus motions can be produced from a single image file. We also analyze the effects on the stimulus of a variety of artifacts: the spatial artifacts produced by the halftoning process, the spurious motion signals produced by the interaction of these spatial artifacts and the temporal animation, the artifacts produced by intensity quantization (and inaccurate gamma correction), and, finally, the important but rarely considered artifact of nonlinear spatial summation. We propose a physical model to account for the observed failures of pixel independence that provides an excellent fit to our measurements.

INTRODUCTION

Motion Stimuli on Digital Displays

Computer graphic displays offer the experimenter a flexible tool for the presentation of visual stimuli. Nevertheless, there are still a number of interesting stimuli that pose technical problems. A particularly demanding class of stimulus consists of slowly moving patterns; for the sake of concreteness, in the following discussion we will focus on the production of a particular stimulus, a sinusoidal grating drifting slowly with a velocity of 0.1 pixel/frame. Such a stimulus might be used in the measurement of spatiotemporal contrast sensitivity, as described by Kelly.[1] The technique described in this paper, however, is applicable to virtually any stimulus composed of drifting gratings, stimuli that are used commonly to investigate motion perception.

Use of Scroll and Pan Registers to Drift an Image

A common feature of many graphics controllers is a set of registers that control the portion of image memory that is displaced. The names used to refer to these registers vary from manufacturer to manufacturer, some examples being window registers, origin registers, and pan and scroll registers. In this paper we refer to them as pan and scroll registers. Regardless of terminology, this hardware feature does the same thing in all units: it allows the image to be displaced laterally (panned) or vertically (scrolled) by an integral number of pixels. When the setting of these registers is synchronized with the display refresh, smooth motion can be generated.

Digital quantization of position can cause a problem, however, in the production of moving stimuli on digital displays. The quantization problem arises when one wants to generate velocities that are not an integral number of pixels per frame time. Consider again the problem of drifting an image at a rate of 0.1 pixel/frame. If the desired displacement for any given frame is simply rounded to the nearest integer and then applied to the pan and scroll registers, the result will be jerky motion: instead of moving smoothly with a velocity of 0.1 pixel/frame, the display will jump by one pixel every 10 frames. For normal frame rates, this is usually unacceptable.

A trick may be used if the stimulus is a periodic one, such as a sinusoidal grating. Let us assume that we wish to drift a grating having a period of N pixels, for some integer N. To use the trick, we instead must force the spatial period to have a particular nonintegral value that is slightly different from that desired. If the desired velocity is 0.1 pixel/frame, we would approximate the desired grating period of N with $N + 0.1$. Since this new period is not an integer, the resulting digital image will have a period of $10N + 1$ pixels. (We assume that we have enough memory to contain at least two periods of $10N + 1$ pixels.) Now consider resampling the same grating shifted by 0.1 pixel. The resulting image is equivalent to the original shifted by N pixels! Thus we can produce a smoothly drifting grating by computing a single image and setting the scroll register to 0, N, $2N$, ..., $9N$, $10N$, $N - 1$, $2N - 1$, $3N - 1$, and so on. Thus we see that for periodic patterns we can generate smooth motion from just a single image. The spatial frequency cannot be specified arbitrarily, however, and to vary velocity we must compute and load a different image.

All techniques involving scrolling require that we have extra display memory so that there is new image to move into the display area as we scroll. (Some units provide a wrap-around feature, but this is useless if the pattern does not match up at the edges.) Another problem shared by all techniques involving scrolling is that they are unsuited to displaying moving patterns whose contrast is modulated by a stationary window having various gradations of contrast. A common example of such a stimulus is a Gabor packet: this refers to a sinusoidal grating windowed by a Gaussian. (One of the first of many experiments using such a stimulus was that described by Watson and Robson.[2]) We can imag-

0740-3232/89/081217-11$02.00

ine drifting the grating while keeping the Gaussian envelope stationary; this vignetting is often done to eliminate edge effects. This clearly cannot be done by scrolling, since that would displace both the grating and the envelope.

The Brute Force Solution: Image Sequencing

The brute force solution is to compute a new image for each frame desired in the sequence and to load the complete set of images into the display memory before the presentation. The image sequencing is then accomplished by modifying pan and scroll registers on a frame-by-frame basis. Unlike the scrolling technique described in the preceding subsection, there is no overlap between successive frames. Unfortunately, this technique is rarely practical, even for systems in which the amount of display memory is adequate. For two-dimensional stimuli, the amount of computation per image is generally large, so it is generally not feasible to compute individual stimuli during the course of an experimental session on a trial-by-trial basis. A better approach is to compute and store all the images before running an experiment. This may allow the experiment to run at an acceptable rate, but if even a moderate number of different stimuli are needed the large requirements for disk storage space may rule out this approach. For example, to store a 10-frame sequence at a spatial resolution of 512×512 requires 2.5 megabytes. Although this amount in and of itself may not sound so large, for an experiment this must be multiplied by the number of different stimuli to be presented.

When the number of images to be sequenced is small (or the images are small), image sequencing may be feasible and is the most straightforward approach. In the context of our hypothetical problem, we can imagine generating 10 images, each representing the original scene but resampled at locations differing by 0.1 pixel. If the system has enough memory to hold all 10 images, they can be preloaded and then sequenced. The cycle of 10 images thus depicts smooth motion over a distance corresponding to 1 pixel; if vignetting is not necessary, then the cycle can then be repeated with everything shifted over by 1 pixel (using pan and scroll registers as described in the preceding subsection) to produce a longer sequence. The utility of this approach is limited by the amount of display memory, since all 10 frames must be preloaded. For one-dimensional stimuli that do not need to be vignetted along their constant dimension, a hardware enlargement, or zoom, feature often incorporated into bit-mapped display devices can reduce the memory requirements of image sequencing. The appearance of a vertical grating will be unaffected by zooming in the vertical dimension; thus the memory requirements for each frame can be reduced by the maximum zoom factor.

Color Table Animation

A common feature on modern raster graphics systems is a programmable color lookup table (LUT), which is indexed by the pixel value. With this feature, the screen intensity at a particular pixel is proportional not to the numerical value of that pixel but rather to the value of the LUT entry that is indexed by that pixel. The number of LUT entries is generally equal to the number of possible pixel values, although some systems provide multiple LUT's. Thus on a system with 8 bits/pixel, there would be 256 (2^8) LUT entries.

The existence of this hardware feature provides an economical method for producing motion of certain stimuli. Successive frames are produced not by reloading image memory but simply by loading a few LUT entries. Techniques based on this approach are generally referred to as color table animation.[3] For example, suppose that we want to make a line move across the screen in apparent motion. We begin by drawing the line in each of its positions in the same frame but using a different color for each position. (For simplicity, we assume that the different instances do not overlap, although overlap can be accommodated.) We can use the LUT to make this image appear as a single instance of the line simply by programming all the colors but one to have the same intensity values as the background. One color will be programmed to a different value; the line that was drawn in this color will be visible. To make the line jump to the next location, we must change only two color table locations. The technique gains its power from the fact that each color table location potentially affects the entire screen.

This technique can also be applied to stimuli such as gratings. Suppose that we want to drift a vertical grating. We begin by covering the entire screen with vertical lines, using each color (pixel value) in sequence. (The number of available colors must be greater than or equal to the grating period in pixels.) To produce a sinusoidal grating we simply load a series of numbers representing the desired waveform into the LUT. This works because we have filled in the screen so as to establish a direct relationship between LUT indices and screen positions. To drift the grating we reload the LUT with the waveform resampled appropriately.

In the above example, there is nothing special about the sinusoidal waveform. In fact, we can choose any set of waveforms, and each may be drifted by a different amount on each frame. Since the total amount of data is relatively small, it may be feasible to compute the resampled waveforms in real time. Alternatively, since the storage requirements are not severe, the data can be precomputed.

This technique is quite powerful when the desired stimuli are one dimensional and periodic. The technique is less suitable for two-dimensional patterns, such as grating compounds having components differing in orientation (plaids). In the one-dimensional case, the maximum period is equal to the number of distinct pixel values. For plaids, the spatial pattern is periodic in two dimensions, so the same finite set of pixel values must be distributed over the two-dimensional unit cell. This reduces the maximum linear period; for example, when the components are equal in frequency and perpendicular, the maximum period is 16 pixels. This limitation forces the user either to restrict the stimuli to high spatial frequencies or to reduce the effective spatial resolution of the display by using clumps of pixels having the same value.

The Problem of Intensity Quantization

We have seen that the quantization of position imposed by the pan and scroll registers complicates the task of producing small movements. In a like fashion, the finite resolution of the digital-to-analog converters (DAC's) that produce the video signal complicates the task of presenting a stimulus at low contrast. For example, with a system having a DAC resolution of 8 bits, a change of 1 least-significant bit about the mean level will produce a luminance change of approximately 1%. Since under optimal conditions the human visu-

al system can detect contrasts as low as 0.5%, this finite DAC resolution will introduce noticeable quantization errors into a low-contrast sine-wave grating. For many systems, this makes the color-cycling technique described in the preceding subsection inadequate for displaying patterns at low contrasts.

Stimuli such as sine-wave gratings that must be presented at low contrast but with good gray-level resolution are often generated by using analog equipment, to avoid the problem of intensity quantization that is present with digital systems. Several methods have been used to overcome the problem of intensity quantization in digital display systems. One approach, described by Watson *et al.*,[4] is to use an analog mixer to combine two (color) channels to produce a single monochrome channel with enhanced intensity resolution. While this solution is satisfactory for many monochrome applications, it cannot be generalized easily for color display.

Another class of solutions enhances intensity resolution at the expense of redundant spatial resolution. Reproduction of continuous-tone photographs in newspapers is accomplished by one such technique, commonly referred to as halftoning. A large variety of algorithms exist for halftoning digital images; these were reviewed by Stoffel and Moreland[5] and Ulichney[6]; see also the recent paper by Knuth.[7] Mulligan[8] has analyzed some of the problems particular to the halftoning of color images.

All halftoning schemes gain intensity resolution at the expense of introducing some high-frequency spatial noise. The contrast of this noise is proportional to the pixel contrast (i.e., the contrast between the light and dark halftoning elements) and therefore is reduced at low signal contrasts, assuming that the signal contrast is varied efficiently by changing the pixel contrast as opposed to redoing the halftoning at the new signal contrast. Producing a constant ratio between the signal and the noise for a given halftone is distinct from the more traditional scheme of setting each pixel independently to the closest approximation to the desired luminance at that point; in the latter case the quantization noise (at harmonics of the signal frequency) appears only at low signal contrast. The fact that the quantization noise in halftoned patterns is not concentrated at harmonics of the signal frequency and does not change its character with signal strength makes these patterns particularly useful for experiments requiring the presentation of stimuli at many different (and often low) contrasts (i.e., threshold measurement).

METHOD

Requirements

We seek a technique for producing motions that satisfies the following criteria: (1) the speed is continuously variable, (2) speed changes may be made without reloading image memory, (3) gratings may be drifted in stationary windows, and (4) presentations may be made for arbitrary durations. The remainder of this paper is devoted to a discussion of a technique that satisfies all these criteria. Although every attempt has been made to keep the discussion independent of any particular graphics system, the following assumptions must be made: (1) the image memory consists of 2 or more bits per pixel and (2) the final screen intensity is not necessarily proportional to the pixel value but is determined by the contents of a programmable LUT that is indexed by the pixel value. It is also necessary that the LUT be programmable on a frame-by-frame basis. Since the amount of data is never large (compared with the amount of data in an image), this is not usually limited by input–output rates.

The number of bits allocated to each entry in the hardware LUT is closely related to the number of bits used by the video DAC's. For a monochrome system, these numbers will be the same. For a color system, the number of bits in each LUT entry will be 3 times the number of individual DAC bits; this is often expressed in bits per phosphor. The number of entries must be 2^n, in which n is the number of bits per pixel. A possible configuration for a color system consists of image memory having 8 bits/pixel and 8-bit DAC's for red, green, and blue. In this case, the LUT would have 256 (2^8) entries of 24 (3×8) bits each, corresponding to 8 bits/phosphor.

In the following subsections, we frequently discuss setting the LUT data to a particular set of values. It is important to realize that typical display monitors exhibit a highly nonlinear relationship between input voltage and output light intensity (often referred to as the gamma nonlinearity). This accelerating nonlinearity can often be described well by a power function with an exponent between 2 and 3. Video broadcast standards prescribe a certain value of the exponent, but individual monitors must be calibrated for precision work. Therefore, if one wishes to display an intensity of exactly half of the maximum luminance, the corresponding LUT value will not be simply half of the maximum numerical LUT setting but must be a particular value determined from the calibration data. The process of determining the appropriate LUT value for a particular desired luminance is referred to as gamma correction. It is often convenient from a programming standpoint to embed the gamma correction process in a low-level subroutine so that the transformation is done transparently to the user; that is, the user specifies a desired luminance, and the system automatically determines the appropriate LUT value. In the following discussion the values that we refer to as LUT data represent desired luminances; these would correspond to the actual numbers entered into the hardware LUT only if the monitor is linear (no gamma correction required).

Description of the Method

To understand the technique, it is necessary to realize that any display that has more than 1 bit/pixel can be thought of as consisting of a number of bit planes, in which each bit plane is thought of as an independent bit-map image. (In the following discussion we will assume that set bits correspond to bright pixels.) Consider a system with two bit planes (2 bits/pixel); the corresponding LUT will have 4 entries. If each bit plane contains a different bit-map image, we display the image in a single bit plane, say, bit plane 0, by programming the LUT as follows:

LUT index		LUT data	
0	(00_2),	$l_{00} = 0$	(black),
1	(01_2),	$l_{01} = 2L_0$	(white),
2	(10_2),	$l_{10} = 0$	(black),
3	(11_2),	$l_{11} = 2L_0$	(white).

The numbers in parentheses to the right of the indices are the binary representations of the corresponding indices. This representation is useful because each binary digit of the

LUT index represents a different bit plane. In the above example, LUT entries are set to white if bit 0 (the least-significant bit) is set, and they are set to black otherwise. I_0 represents the luminance of the display's middle gray. Let us introduce the symbol l_{ij} to represent the table entry whose index has digits i and j when expressed in binary form. We can then represent the pattern in the table more succinctly as follows:

$$l_{ij} = I_0[1 + (-1)^{i+j}]. \quad (1)$$

To display the image in bit plane 1, on the other hand, we must set the LUT data to white whenever bit 1 of the index is set and to black otherwise. The compact description of this relationship is similar to Eq. (1):

$$l_{ij} = I_0[1 + (-1)^{j+i}]. \quad (2)$$

We see that by programming the LUT appropriately, we can display (or conceal) the bit-map images in any single bit plane. We can generalize this principle to produce displays whose luminances are sums of the patterns in the two bit planes. The following settings of the LUT produce a display corresponding to a modulation of the image in bit plane 0 at contrast C_0 added to a modulation of the image in bit plane 1 at contrast C_1:

LUT index	LUT data
$0\ (00_2)$,	$l_{00} = I_0(1 - C_0 - C_1)$,
$1\ (01_2)$,	$l_{01} = I_0(1 + C_0 - C_1)$,
$2\ (10_2)$,	$l_{10} = I_0(1 - C_0 + C_1)$,
$3\ (11_2)$,	$l_{11} = I_0(1 + C_0 + C_1)$.

Alternatively,

$$l_{ij} = I_0[1 + (-1)^{i+1}C_0 + (-1)^{j+1}C_1]. \quad (3)$$

Since none of the l_{ij} can be negative, the sum of the contrasts $C_0 + C_1$ must be less than or equal to 1.

Consider once again the problem of producing a drifting grating. An elementary trigonometric identity tells us that a drifting grating may be decomposed as the superposition of two counterphase-modulated gratings, in quadrature spatial and temporal phase:

$$\sin(\omega x + \nu t) = \sin(\omega x)\cos(\nu t) + \cos(\omega x)\sin(\nu t).$$

We can use this identity in the following manner: first, we reduce the sine and cosine (spatial) phase gratings to 1-bit/pixel images by halftoning. The method does not depend on which particular halftoning algorithm is used; in our laboratory we use a variant of error diffusion[9] described by Mulligan.[8] This algorithm differs from the original error-diffusion algorithm in that the lines are scanned alternately from left to right and from right to left in order to eliminate structured artifacts. These two halftoned images are then loaded into two bit planes of the display device. A drifting grating can now be displayed by reloading the LUT on a frame-by-frame basis: we rewrite Eq. (3), making C_0 and C_1 functions of time as follows:

$$l_{ij} = I_0[1 + (-1)^{i+1}C\cos(\nu t) + (-1)^{j+1}C\sin(\nu t)], \quad (4)$$

The parameter C is simply the contrast of the drifting grating. At this point we recall that earlier in this section we observed that the sum of the individual bit-plane contrasts must be less than or equal to 1; in this case, the sum is maximal when $\nu t = \pi/4$, for which it reaches a value of $\sqrt{2}C$. The maximum contrast at which we can drift a grating is therefore $\sqrt{2}/2$, or approximately 71%. [If it is imperative to achieve higher contrasts, this can be done by allocating more bit planes to other phases of the grating besides 0 and 90 deg. If we produce multiple versions of the grating, in which successive instances have a relative phase difference of ϕ degrees, then the maximum attainable drifting contrast will be $\cos(\phi/2)$.]

This scheme has several advantages. Although there is a substantial amount of computation in generating the halftoned images, once that has been done a single image may be loaded that can produce stimuli of many different contrasts and drift velocities. Windowing the drifting gratings with a stationary Gaussian envelope (or any other envelope) can also be done; the product of the window and the grating is taken before the halftoning step.

In the discussion above, we assumed the existence of only two bit planes. If more are available, however, additional gratings (possibly differing in spatial frequency and orientation) may be loaded into additional pairs of bit planes. Thus this scheme is admirably suited to the generation of sine-wave plaid patterns.[10-16] Since the speeds of the various grating components are independent, a variety of resultant plaid velocities may be generated from a single base image. Furthermore, since the contrasts of the various plaid components are similarly independent, the effect of the relative contrast of the components may be studied without the need to generate new images for each stimulus.

Shadlen *et al.*[17] demonstrated an interesting technique for producing nonrigid, constant-temporal-frequency motion based on flickering arbitrary base images in temporal quadrature with spatial quadrature image partners obtained by applying odd-symmetric filter functions. Our technique is eminently suited for this type of presentation also; in this case the sine and cosine spatial phase gratings are simply replaced by the original and filtered images.

LIMITATIONS

We have described a method of combining digital halftoning with dynamic LUT programming to produce drifting plaid patterns. We have found this method to be superior to other approaches in both flexibility and freedom from artifacts. Nevertheless, there are a few artifacts that are unique to this technique, and the remainder of this paper is devoted to their analysis.

This section is divided into four subsections. In the first subsection we consider the spatial noise introduced by halftoning; this is a cursory treatment of a subject that is covered in depth by Ulichney,[6] but we include it for completeness and to provide a framework for things to come. In the second subsection we analyze what happens to the halftoning noise when the color table is modified to drift the patterns, and we show how the resulting spatiotemporal noise can be decomposed into components of flicker and drift. In the third subsection we analyze artifacts arising from intensity quantization as a result of finite DAC resolution. Finally, we introduce a nonlinear monitor artifact that to our knowledge has not heretofore been considered.

We hope that the lengthy discussion of artifacts does not cause the reader to disparage the utility of the technique; our intention is to demonstrate that the artifacts are easy to calculate, so that researchers using the method can assure themselves that the artifacts in their stimuli are negligible.

Purely Spatial Artifacts Introduced by Halftoning

Halftoning increases intensity resolution at the expense of spatial resolution. Thus the technique described above is unsuitable for applications requiring high spatial frequencies. If large numbers of spatial cycles are not required, however, high retinal spatial frequencies can be obtained with a larger viewing distance or optical minification.

The halftoning process also adds noise. Figure 1 shows a unit-contrast Gabor patch halftoned on a 256 × 256 array of pixels, with a carrier period of 32 pixels and a two-dimensional Gaussian window having a standard deviation of 45.3 pixels. This test image was used for all the following noise computations. An error signal was computed by subtracting the original (unquantized) image from the halftoned version of the image. The discrete Fourier transform (DFT) of this error signal was then computed, after first multiplying with a two-dimensional Gaussian window function with a standard deviation of 33.9 pixels to reduce spectral leakage and wraparound artifacts.[18] A radially averaged spectrum was computed from the two-dimensional DFT by the method of Ulichney.[6] This is shown in Fig. 2(a), which illustrates the high-pass character of the halftoning noise. The horizontal axis represents spatial frequency averaged over all orientations, with 1/128-cycle/sample bins. The vertical axis represents the logarithm to the base 10 of the average Fourier amplitude. The abscissa extends beyond the nominal Nyquist frequency of 0.5 cycle per sample by a factor of ~1.4 ($\sqrt{2}$). Frequencies above 0.5 cycle per sample are not present at all orientations; the highest frequency represents the corner of the two-dimensional frequency space and contains only samples from orientations of 45 and 135 deg.

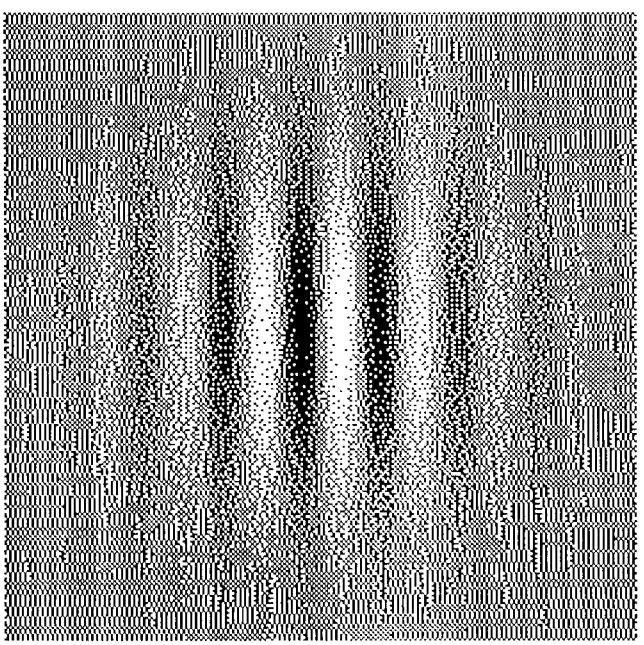

Fig. 1. Two-dimensional Gabor patch halftoned on a 256 × 256 pixel grid. This quantization noise in the image was used to compute subsequent noise spectra.

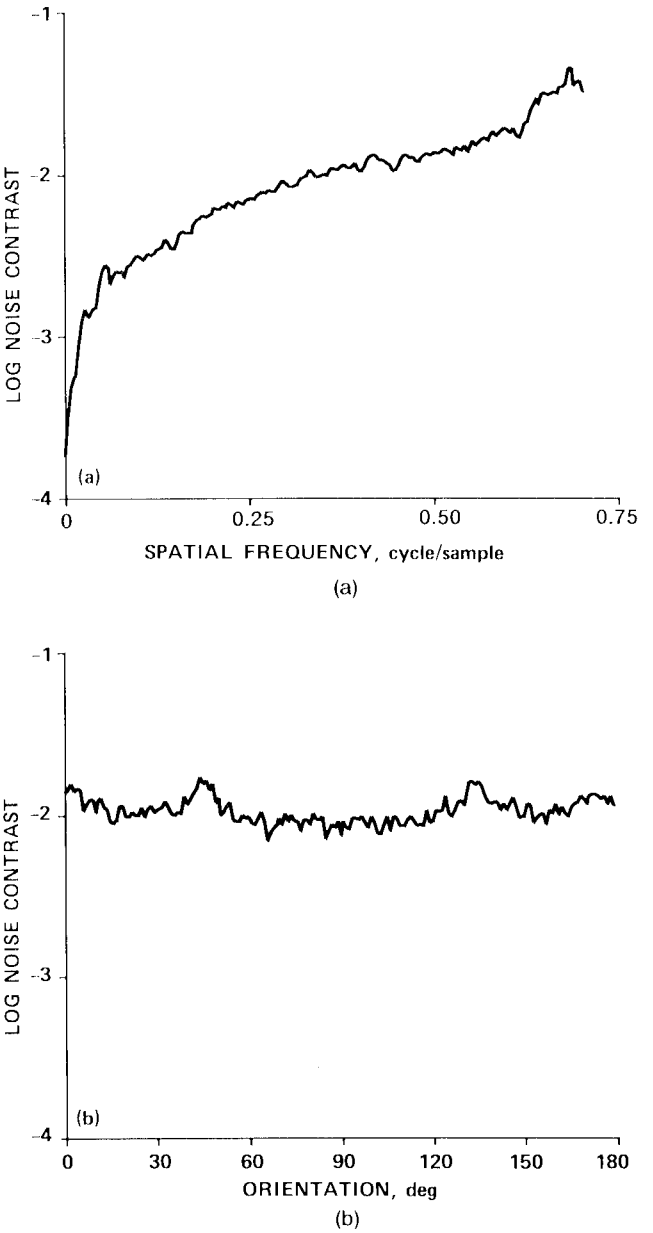

Fig. 2. (a) Radially averaged spectrum of the quantization noise present in the image in Fig. 1. The frequencies above the Nyquist limit of 0.5 cycle/sample are present only at oblique orientations. (b) Similar to (a) but here the two-dimensional noise spectrum was averaged over spatial frequency and is shown as a functon of orientation. The bumps at 45 and 135 are due to the high-pass nature of the spectrum shown in (a) combined with the fact that the highest frequencies are present only at oblique orientations, i.e., the corners of the square spectrum.

In Fig. 2(b), the same DFT has been reduced to a one-dimensional plot, in which instead of averaging over orientation for a fixed spatial frequency we have averaged over all spatial frequencies at a given orientation (1-deg bins). Figure 2(b) shows that there are no systematically oriented structures in the noise with the exception of the two bumps that appear at 45 and 135 deg. These features can be understood in light of the observation made in the preceding

paragraph that the highest spatial frequencies occur only at the oblique orientations; this result, coupled with the data in Fig. 2(a) showing that these high frequencies have the highest average amplitude, explains the small bumps seen in Fig. 2(b).

The halftoning procedure partitions the quantization noise selectively into the high picture frequencies. The viewing distance may be adjusted so that this noise corresponds to high retinal frequencies to which the human visual system is relatively insensitive. Furthermore, when the signal contrast is reduced, the noise amplitude is reduced proportionately. [The curves in Figs. 2(a) and 2(b) are for a signal contrast of 1.0.] Thus in many situations the noise can be made invisible simply by using moderate to low contrasts and high-frequency signals (which permit corresponding large viewing distances, reducing the pixel size).

Spatiotemporal Artifacts

The motion stimulus that we have described is produced by summing sine and cosine spatial patterns that are counterphase modulated in temporal quadrature. We have also shown how this can be accomplished by halftoning the sine and cosine spatial patterns and loading the resulting bit maps into individual display bit planes. In the preceding subsection we have analyzed the static noise in a single bit map. In this subsection we analyze the spatiotemporal characteristics of the noise when we perform the temporal modulation needed to produce the motion stimulus.

Consider a particular spatial noise component in the cosine phase bit map. It has a particular spatial frequency and orientation; for simplicity we will consider it to be a function of a single spatial coordinate (x) and choose our system of units so that the angular frequency has a value of 1. We can then express this particular component as follows:

$$N_c(x) = A_1 \cos(x + \theta_1),$$

in which the parameter A_1 represents the Fourier amplitude and θ_1 is the spatial phase. When we animate the motion sequence, this pattern will be counterphase modulated in sine phase:

$$N_c(x, t) = A_1 \cos(x + \theta_1)\sin(t).$$

We have arbitrarily chosen a unit angular temporal frequency. This counterphase grating can be decomposed as the sum of rightward- and leftward-drifting gratings:

$$N_c(x, t) = \frac{A_1}{2}\,|\sin(t + x + \theta_1) + \sin(t - x - \theta_1)|.$$

The corresponding noise component in the sine phase bit map has some other amplitude and phase, A_2 and θ_2. We can write a similar expression for the spatiotemporal signal contributed by this bit map:

$$N_s(x, t) = A_2 \cos(t)\cos(x + \theta_2)$$

$$= \frac{A_2}{2}\,|\cos(t + x + \theta_2) + \cos(t - x - \theta_2)|.$$

The noise in the composite image is simply the sum of these two quantities:

$$N_c(x, t) + N_s(x, t) = A_R \cos(x + t - \phi_1) + A_L \cos(x - t - \phi_2), \qquad (5)$$

where

$$A_R = \tfrac{1}{2}[A_1^2 + A_2^2 + 2A_1A_2 \sin(\theta_1 - \theta_2)]^{1/2},$$

$$A_L = \tfrac{1}{2}[A_1^2 + A_2^2 - 2A_1A_2 \sin(\theta_1 - \theta_2)]^{1/2},$$

$$\phi_1 = \tan^{-1}\left(\frac{A_1 \cos\theta_1 - A_2 \sin\theta_2}{A_1 \sin\theta_1 + A_2 \cos\theta_2}\right),$$

$$\phi_2 = \tan^{-1}\left(\frac{-A_1 \cos\theta_1 - A_2 \sin\theta_2}{-A_1 \sin\theta_1 + A_2 \cos\theta_2}\right).$$

In Eq. (5) we have expressed a spatiotemporal artifact as a sum of leftward- and rightward-drifting gratings, having independent amplitudes and starting phases. We now propose that this quantity can be considered to be the sum of a stationary flickering component with an amplitude min (A_R, A_L) and a drifting component with an amplitude $|A_R - A_L|$, which moves to the right if $A_R > A_L$ and to the left if $A_R < A_L$.

This decomposition is illustrated graphically in Fig. 3. The original amplitudes and phases can be considered to be polar coordinates of vectors representing the two spatial components. To the right of the two initial vectors is shown the decomposition into rightward- and leftward-drifting components that is described above. Below this is shown a graphical interpretation of the decomposition into flicker and drift. The shorter of the two original vectors (which we assume to be v_2) may be decomposed into parallel and perpendicular components relative to v_1. The perpendicular component can be paired with a fraction of v_1 having equal length to produce a pure drift signal. The remainder of v_1 and the parallel component of v_2 represent gratings having

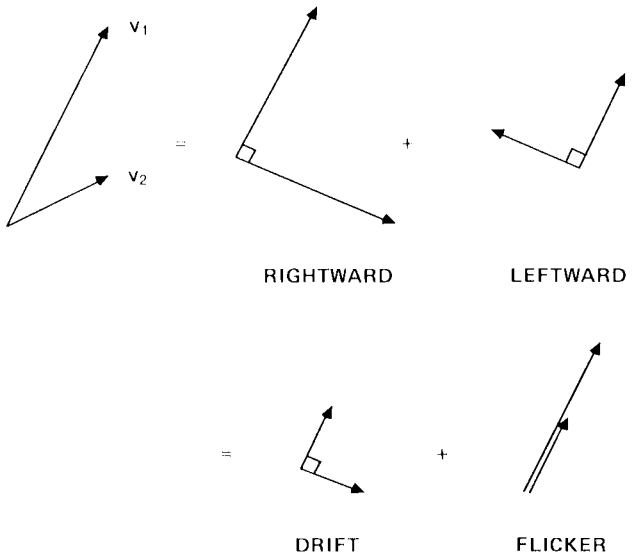

Fig. 3. Graphical construction showing how spatiotemporal noise at a particular frequency and orientation may be decomposed into a sum of drifting and flickering components. The two vectors at the upper left represent the amplitudes and phases of a particular noise component in the halftoned versions of the sine and cosine signal phases. To the right of these vectors is shown the decomposition into rightward and leftward components described in the text. Below is shown an alternative graphical decomposition. The shorter of the two vectors ($\mathbf{v}_2$) is decomposed into parallel and perpendicular components (with respect to $\mathbf{v}_1$); a fraction of the longer vector ($\mathbf{v}_1$) equal in length to the perpendicular component combines with it to form a pure drift component. The remaining parallel components correspond to a pure flicker signal. (The vector lengths shown in the figure are not represented precisely.)

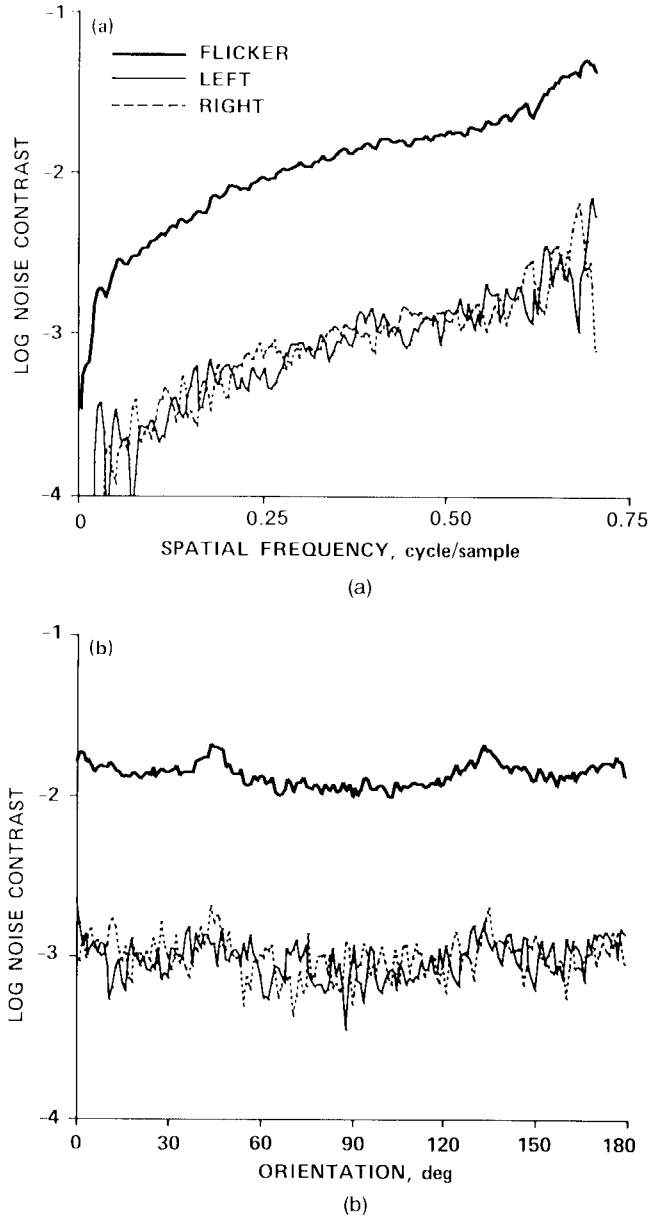

Fig. 4. (a) Radially averaged noise spectra after decomposition into flicker and drift components when the Gabor signal in Fig. 1 is made to drift. The bold curve represents the average flicker amplitude, and the light and dashed curves represent leftward and rightward motion signals, respectively. (b) The same two-dimensional spectra used to generate (a) are averaged over spatial frequency and plotted as a function of orientation.

the same spatial phase; when they are temporally modulated the result is pure flicker.

We estimated the amounts of these types of artifact for the pattern shown in Fig. 1 and its companion cosine phase bit map. Complex Fourier spectra were first computed for the quantization noise in each pattern. For each pair of corresponding Fourier components drift and flicker amplitudes were computed as described above. These were then averaged over spatial frequency [Fig. 4(a)] and orientation [Fig. 4(b)]. The curves for both flicker and drift have roughly the same shape as the curves in Figs. 2(a) and 2(b). The figures show that at a given spatial frequency or orientation the average flicker amplitude is approximately 1 log unit greater than the average amount of rightward or leftward drift, with the drift component evenly partitioned between rightward and leftward components. For oblique orientations, velocity vectors in quadrants 1 and 4 were classified as rightward, those in quadrants 2 and 3 as leftward.

Artifacts Introduced by Intensity Quantization

A conventional method for producing a grating is to produce tonal gradations by using the LUT settings to modulate a series of uniformly drawn strips. Producing a sinusoidal grating using this method depends critically on accurate compensation for the monitor's light-intensity-versus-voltage function, or gamma. Deviations from perfect linearity, caused by either calibration inaccuracy or quantization errors, will produce harmonic distortion of the waveform.

Halftoned gratings, on the other hand, depend on linearity of spatial interactions in the monitor and in the eye to obtain a faithful rendition with no harmonic distortion. Errors in luminance settings, whether due to quantization or to calibration error, give rise to a different type of artifact under these circumstances, which we will investigate in this subsection. When these errors occur as a result of DAC quantization, they may be reduced to some extent by temporal error diffusion, a process analogous to the halftoning that we have done in the spatial domain.

First let us consider the densities of various pixel values in the image. We assume that each of the gratings was halftoned to produce 100% modulation of density of its particular bit. We define $g_1(x, y)$ to represent the local probability of a bit's being set in bit plane 0. This is not an accurate statistical model of an image halftoned by error diffusion, since the state of a particular bit depends on the states of its neighbors. Nevertheless, the function $g_1(x, y)$ does accurately describe the density of set pixels averaged over an area where $g_1(x, y)$ is relatively uniform. Let $g_2(x, y)$ represent the corresponding probability function of the image in bit plane 1. We make the further assumption that the probability of a pixel's having both bits set is simply the product of the probabilities for the individual bits. We introduce the symbol d_{ij} to represent the local probability of a pixel's having the value l_{ij}. We can now write expressions for these probabilities in terms of the probabilities in the individual bit planes:

$$d_{00}(x, y) = [1 - g_1(x, y)][1 - g_2(x, y)],$$

$$d_{01}(x, y) = g_1(x, y)[1 - g_2(x, y)],$$

$$d_{10}(x, y) = [1 - g_1(x, y)]g_2(x, y),$$

$$d_{11}(x, y) = g_1(x, y)g_2(x, y).$$

For the case of vertical gratings, $g_1(x, y) = [1 + \cos(x)]/2$ and $g_2(x, y) = [1 + \sin(x)]/2$. We can therefore simplify the expressions for the functions d_{ij}:

$$d_{00}(x, y) = \tfrac{1}{4}(1 - \sin x - \cos x + \sin x \cos x),$$

$$d_{01}(x, y) = \tfrac{1}{4}(1 - \sin x + \cos x - \sin x \cos x),$$

$$d_{10}(x, y) = \tfrac{1}{4}(1 + \sin x - \cos x - \sin x \cos x),$$

$$d_{11}(x, y) = \tfrac{1}{4}(1 + \sin x + \cos x + \sin x \cos x).$$

Now let us consider setting the LUT with a set of four arbitrary values, $\{l_{ij}\}$. The local luminance (computed over a

neighborhood of a few pixels) is the sum of the light contributed by each type of pixel; this is simply the product of the luminance of that pixel type, l_{ij}, and the density of that pixel type, d_{ij}:

$$L(x, y) = \sum_{i=0}^{1} \sum_{j=0}^{1} l_{ij} d_{ij}(x, y). \tag{6}$$

Let us make the definitions

$$L_{\text{mean}} = \frac{1}{4} \sum_{i=0}^{1} \sum_{j=0}^{1} l_{ij}, \tag{7}$$

$$A_1 = \frac{1}{4} \sum_{i=0}^{1} \sum_{j=0}^{1} l_{ij}(-1)^{i+1}, \tag{8}$$

$$A_2 = \frac{1}{4} \sum_{i=0}^{1} \sum_{j=0}^{1} l_{ij}(-1)^{j+1}, \tag{9}$$

$$A_{\text{prod}} = \frac{1}{4} \sum_{i=0}^{1} \sum_{j=0}^{1} l_{ij}(-1)^{i+1}(-1)^{j+1} \tag{10}$$

and the auxiliary definitions $C_1 = A_1/L_{\text{mean}}$, $C_2 = A_2/L_{\text{mean}}$, and $C_{\text{prod}} = A_{\text{prod}}/L_{\text{mean}}$; we can write Eq. (6) for $L(x, y)$:

$$L(x, y) = L_{\text{mean}}(1 + C_1 \sin x + C_2 \cos x + C_{\text{prod}} \sin x \cos x). \tag{11}$$

This equation tells us that the display contains a sine phase grating with contrast C_1, a cosine phase grating with contrast C_2, and a product grating with contrast C_{prod}. Remembering that $\sin x \cos x = \frac{1}{2} \sin 2x$, we can see that the so-called product term produces an artifact at the second harmonic of the grating.

Normally we will program the $\{l_{ij}\}$ according to Eq. (4); if we could do this exactly, C_{prod} would be zero. Because we are working with a digital system, the desired values of $\{l_{ij}\}$ must be approximated from a finite set corresponding to the possible DAC settings. The quantization errors in the present actual values of the $\{l_{ij}\}$ can cause C_{prod} to be nonzero as well as introduce errors into all the other quantities defined in Eqs. (7)–(10).

We see that in this situation, intensity quantization can introduce errors into the contrasts of the gratings. If there are only two possible gray levels, then a given bit map can be presented only at a single contrast. Ignoring for the moment the problem of gamma correction, if we have N bits of DAC resolution (yielding 2^N gray levels), then we should have $2^N - 1$ possible contrasts if we restrict the mean luminance to the midpoint of the range. In our laboratory we are fortunate to have a display system with 10-bit DAC resolution, and we have not found it necessary to do anything special to compensate for contrast quantization. On systems having lower DAC resolution, the problem of contrast quantization might be alleviated by performing temporal error diffusion. By this we mean that in each frame we desire to present a certain contrast of each component. This translates into a specific set of LUT data values. Intensity quantization limits the precision with which we can approximate any desired set of values. At each frame we compute a contrast error as the difference between the desired contrast and that actually obtained after roundoff:

$$C_{\text{error}}(t) = C_{\text{actual}}(t) - C_{\text{desired}}(t).$$

For the next frame, we substract this error from the desired value before rounding:

$$C'_{\text{desired}}(t + \Delta t) = C_{\text{desired}}(t + \Delta t) - C_{\text{error}}(t).$$

In spatial halftoning, we approximate a gray level by a series of closely spaced transitions between black and white. Likewise, in the case of temporal error diffusion, we approximate a given desired contrast level by a rapid temporal alternation of bracketing contrasts.

Because of the way in which we are synthesizing a drifting grating, a contrast error of the sine or cosine phase component will result in a position error of the composite grating. Because of this, it might be desirable to minimize the error in the contrast ratio of the two components rather than the errors in the individual contrasts themselves.

Although we have used the example of diffusing the error into the desired contrasts, since the errors are actually introduced at the level of the individual LUT entries the various types of contrast error are not independent. Special schemes might be devised to partition the error selectively into the four components identified in Eqs. (7)–(10); for example, it should be possible to minimize the amount of second-harmonic artifact by permitting larger quantization errors on the grating component contrasts. Such a scheme might be performed by using different temporal spread functions for the various error components. For example, if for some reason we could not tolerate any second-harmonic artifact, we could subtract all the error in C_{prod} in the subsequent temporal sample (as was suggested above) while partitioning the other errors into fractions that would trickle in over a number of temporal samples. This type of approach was applied by Mulligan[8] to the halftoning of color images with the goal of partitioning the chromatic components into a lower-frequency band than the luminance errors.

Artifacts Due to Nonlinear Spatial Interactions

An implicit assumption underlying the method described above has been that the total luminous flux contributed by a given pixel depends only on its numerical value (and the LUT settings, of course) and is independent of the values of neighboring pixels. (It is acceptable for pixels to overlap somewhat, as long as their luminances summate linearly.) Unfortunately, many monitors fail in this respect. Figure 5 shows some empirical data characterizing the nature of this failure. (We do not name the make and model as a courtesy to the manufacturer; in all fairness, we suspect that deviations of this magnitude may be present in many so-called good monitors.) In Fig. 5 we have plotted space-average luminance as a function of pixel density while holding pixel contrast constant at a value of 1.0 (a pixel contrast of 1.0 means that all pixels were either black or white). Each point represents a reading made from a different two-color bit-map image; these were obtained simply by applying the halftoning algorithm to a uniform field of variable level. Figure 5 shows that the luminance is not simply proportional to the density of white pixels; regions containing a mix of black and white pixels are dimmer than they should be, with the deviation being largest when the density is near 0.5 (where black and white pixels alternate on a pixel-by-pixel basis).

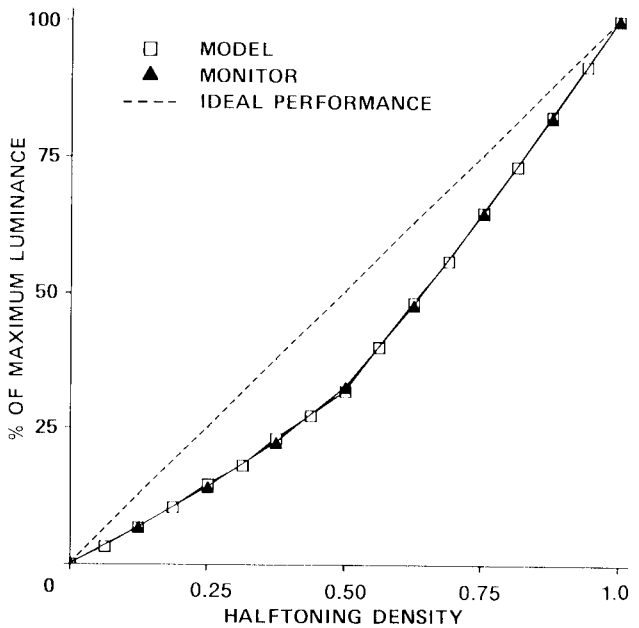

Fig. 5. Space-average monitor luminance, plotted as a function of halftone density. Pixel contrast was held constant at a value of 1.0. The straight line with a slope of 1 represents ideal performance. The actual data (filled triangles) show a significant deviation. Open squares show predictions from a model consisting of an exponential low-pass filter with a time constant of 0.65 pixel followed by the monitor gamma nonlinearity modeled as a power function with an exponent of 2.6.

One type of failure that we fortunately do not observe in this particular monitor can occur in a monitor that has an inadequate power supply. In this case the luminance of a large white area will be less than that of a smaller white area because the power supply droops under the load imposed by a large illuminated area. A defect such as this would show up as a compressive nonlinearity in a measurement of luminance versus pixel density. This type of nonlinearity is the opposite of that which we observe with our monitor, as is shown in Fig. 5.

What then might be the mechanism responsible for the accelerating nonlinearity that we observe? The clue that led to our hypothesis is that the monitor gamma (the function relating input voltage to output luminance) is also a positively accelerating function. This in and of itself should have no effect, since all our bit maps are composed of only two types of pixel. The gamma function might affect the actual luminances corresponding to black and white, but our idealized conception of the gamma function is that of a point nonlinearity with no spatial interaction. We can produce a spatial interaction, however, by postulating that the incoming video signal is passed through a low-pass filter before the nonlinearity. Without a detailed study of monitor electronics this seems to be a plausible hypothesis; the low-pass filter could easily be the first stage of amplification in the monitor circuitry.

In our model, the artifact illustrated in Fig. 5 would be produced as follows: at the extremes of pixel density, there are few transitions from light to dark, so the low-pass filter has a negligible effect. When there are many transitions, however, the effect of the low-pass filter is to transform a signal that nominally represents black–white–black to one that represents dark gray–light gray–dark gray. After this signal passes through the monitor's gamma nonlinearity, the mean of dark gray and light gray is no longer equal to the mean of black and white. This idea is illustrated graphically in Fig. 6.

We simulated this model by digitally filtering the test images with an exponential filter and then passing it through a point nonlinearity corresponding to the gamma function. The nonlinearity that we used was a power function with an exponent of 2.6, which approximates the calibration data measured with spatially uniform fields. The time constant of the low-pass filter was varied to fit to the data by eye. Because of the nature of the raster scan process, the temporal bandwidth of the video amplifier can be identified with a corresponding space constant in the resulting image on the screen. Model predictions obtained by using a space constant corresponding to 0.65 pixel are shown with the actual data in Fig. 5. The agreement between model and data is quite good, although it should be noted that the value of the corresponding time constant is more than twice as large as the manufacturer's specification of the video amplifier rise time. It may be that the amplifier specifications are correct and that the filtering occurs at some other stage; alternatively, the specifications may be inaccurate. We are also open to the suggestion that the model is inaccurate in a mechanistic sense; this does not concern us, since the purpose of the model is simply to allow us to predict the artifacts in arbitrary stimuli.

An implication of this model is that the deviation from linearity seen in Fig. 5 should depend on pixel contrast. Obviously there is no effect of pixel density when the pixel contrast is 0.0; the model makes a specific prediction, however that the size of the nonlinearity will depend in a nonlinear way on pixel contrast.

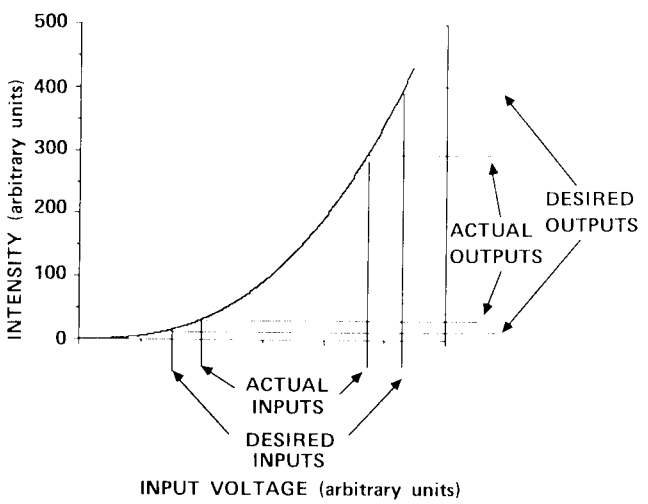

Fig. 6. Diagram illustrating why the model predicts large deviations from linearity at high pixel contrasts. The curve represents the monitor's gamma nonlinearity, in this case a power function with an exponent of 2.6. Below the horizontal axis the actual inputs are shown to be displaced equal amounts from the desired inputs; this is presumed to occur through the action of a low-pass filter somewhere in the monitor electronics. After passing through the gamma nonlinearity, the actual outputs are displaced unequal amounts from the desired outputs, resulting in a shift of the mean luminance. For small pixel contrasts the gamma function is approximately linear, so no deviation occurs.

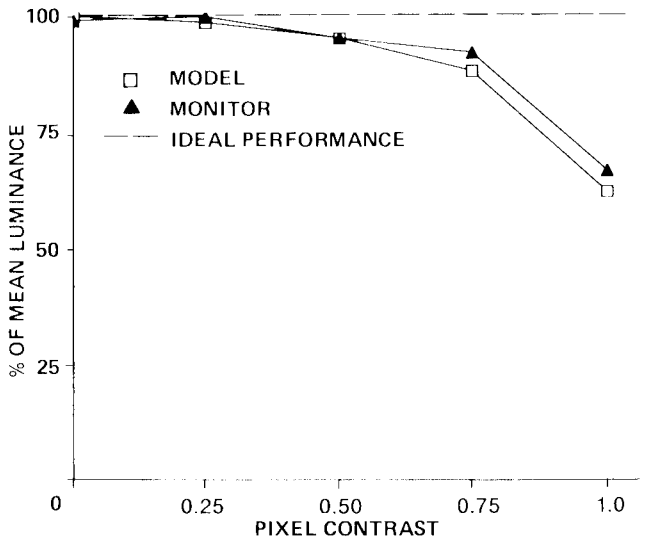

Fig. 7. Space-average monitor luminance is plotted as a function of pixel contrast. Halftone density was held constant at a value of 0.5. Ideal performance would be represented by a perfectly flat curve. Actual measurements (filled triangles) and model predictions (open squares) exhibit large deviations only at the highest pixel contrasts. Input voltages for each pixel contrast were determined by a prior gamma correction calibration done with uniform fields.

Figure 7 shows the results of measurements made to test this idea. For these measurements, only a single bit map was used, having a pixel density of 0.5 (the density at which we observed the greatest departure from linearity). The abcissa represents pixel contrast, and the ordinate represents the measured space-average luminance. The data show that the space-average luminance is relatively constant for a pixel contrast as large as ~0.7, at which point it decreases dramatically. It is important to observe at this point that we attempted to correct for the monitor gamma before programming the values to produce a particular pixel contrast; if we had not done this we would not have expected to get a flat curve, regardless of any spatial interactions.

Figure 7 also shows the model predictions. The predictions shown in Fig. 7 were obtained by using the same model parameters that were obtained by optimizing the fit to the data in Fig. 5. The agreement is quite good, although not quite so good as in Fig. 5. This is probably because the results in Fig. 7 depend more heavily on the precise nature of the gamma function, since gamma correction was applied to the raw input images for both the model and the actual measurements. The model gamma function was a power function with a single exponent, whereas the gamma correction of the actual monitor used a slightly different exponent at the lowest values. Since the extreme linearizing table entries are used only at the higher pixel contrasts, this is the region in which we might expect to find small deviations between the measurements and the model, as are observed.

It is likely that our model oversimplifies the physical processes acting in the monitor electronics. Regardless of the actual mechanism, however, we believe that having a valid descriptive model of the artifact can be of some use in devising strategies to eliminate it. For static images that will be presented at a fixed contrast, a workable solution could be obtained simply by passing the input image through a compensating nonlinearity before halftoning. Unfortunately, this will not work for images in which the pixel contrast is changed dynamically, however, since the magnitude of the artifact depends on pixel contrast (Fig. 7). We have found only one solution to the problem: we used a higher-bandwidth monitor that had only a tiny amount of artifact. Since many monitors may have this defect, however, we must advise potential users of the technique to check their monitors and, if necessary, to reduce the artifact by working at lower pixel contrasts.

The above analysis is only for the case of two pixel species; a complete analysis of the artifacts present in our drifting sine-wave stimulus is beyond the scope of this paper. Based on the good fit between the model and our measurements of the artifact, however, we believe that the model is adequate for estimating the artifacts present in any given stimulus.

CONCLUSION

Halftoning in conjunction with dynamic LUT modification is a powerful technique for the generation of moving stimuli for vision research. By using this technique, contrast and temporal frequency can be varied with a negligible amount of computation once a single base image has been produced. Since only two bit planes are needed to display a single drifting grating, an 8-bit/pixel display can be used to generate four-component plaids, in which each component of the plaid has independently programmable contrast and temporal frequency (speed). (The same display might be used to generate two-component plaids in which the constituent patterns are halftoned to 2 bits/pixel instead of 1, reducing the halftoning noise for high-contrast patterns.) Artifacts can be minimized by careful monitor gamma correction and by working at pixel contrasts at which monitor spatial interactions are linear. This technique makes it possible to produce complex motion stimuli that are difficult if not impossible to produce by other means.

ACKNOWLEDGMENTS

We thank Al Ahumada, Lew Hitchner, and Beau Watson for useful comments on the manuscript. The research of Lee Stone was supported by a National Research associateship.

REFERENCES

1. D. H. Kelly, "Motion and vision. II. Stabilized spatio-temporal threshold surface," J. Opt. Soc. Am. **69**, 1340–1349 (1979).
2. A. B. Watson and J. G. Robson, "Discrimination at threshold: labelled detectors in human vision," Vision Res. **21**, 1115–1122 (1981).
3. R. G. Shoup, "Color table animation," Comput. Graphics **13**, 8–13 (1979).
4. A. B. Watson, K. R. Nielsen, A. Poirson, A. Fitzhugh, A. Bilson, K. Nguyen, and A. J. Ahumada, "Use of a raster framebuffer in vision research," Behav. Res. Methods Instrum. Comput. **18**, 587–594 (1986).
5. J. C. Stoffel and J. F. Moreland, "A survey of electronic techniques for pictorial image reproduction," IEEE Trans. Commun. **C-29**, 1898–1925 (1981).
6. R. Ulichney, *Digital Halftoning* (MIT Press, Cambridge, Mass., 1987).
7. D. E. Knuth, "Digital halftones by dot diffusion," ACM Trans. Graphics **6**, 245–273 (1987).

8. J. B. Mulligan, "Minimizing quantization errors in digitally controlled CRT displays," Color Res. Appl. **11,** S47–S51 (1986).
9. R. W. Floyd and L. Steinberg, "An adaptive algorithm for spatial gray scale," in *SID 1975 International Symposium Digest of Technical Papers* (Society for Information Display, Playa del Rey, Calif., 1975), pp. 36–37.
10. E. H. Adelson and J. A. Movshon, "Phenomenal coherence of moving visual patterns," Nature (London) **300,** 523–525 (1982).
11. J. A. Movshon, E. H. Adelson, M. S. Gizzi, and W. T. Newsome, "The analysis of moving visual patterns," in *Pattern Vision Mechanisms*, C. Chagas, R. Gattas, and C. Gross, eds. (Vatican Press, Rome, 1985) [reprinted in Exp. Brain Res. Suppl. **11,** 117–152 (1986)].
12. V. P. Ferrera and H. R. Wilson, "Direction specific masking and the analysis of motion in two dimensions," Vision Res. **27,** 1783–1796 (1987).
13. V. P. Ferrera and H. R. Wilson, "Perceived direction of moving 2D patterns," Invest. Ophthalmol. Vis. Sci. Suppl. **29,** 264 (1988).
14. F. L. Kooi, R. L. De Valois, and T. K. Wyman, "Perceived direction of moving plaids," Invest. Ophthalmol. Vis. Sci. Suppl. **29,** 265 (1988).
15. L. S. Stone, J. B. Mulligan, and A. B. Watson, "Neural determination of the direction of motion: contrast affects the perceived direction of a moving plaid," Soc. Neurosci. Abstr. **14,** 1251 (1988).
16. L. Welch, "Speed discrimination and the aperture problem," Invest. Ophthalmol. Vis. Sci. Suppl. **29,** 264 (1988).
17. M. Shadlen, T. Carney, and G. Switkes, "Illusory rotation, expansion and contraction from transitions in local symmetry," Invest. Ophthalmol. Vis. Sci. Suppl. **28,** 300 (1987).
18. F. J. Harris, "On the use of windows for harmonic analysis with the discrete Fourier transform," Proc. IEEE **66,** 51–83 (1978).